CLAIMS

What is claimed is:

1. A system to enable queues for COBOL programs, the comprising:

   a memory space;

   an operating system maintaining a key related to an address of the memory space;

   a COBOL routine maintaining the key in an index, the COBOL routine communicating with the operating system to receive the address of the memory space based on the key;

   a COBOL program coupled to receive the memory address based on the key from the index of the COBOL routine.

2. The system of Claim 1, wherein the COBOL program operably calls the COBOL routine using an identifier and wherein the index maintains the identifier associated with the key.

3. The system of Claim 2, wherein the index maintains a plurality of identifiers each associated with one of a plurality of keys maintained by the index.

4. The system of Claim 3, wherein the plurality of identifiers are further defined as an alphanumeric identifier.

5. The system of Claim 3, wherein the plurality of identifiers are further defined as names.

6. The system of Claim 1, wherein the COBOL program receives the memory address via a linkage section of the COBOL program.

7. The system of Claim 1, wherein the COBOL program is operable to receive the memory address to enable the COBOL program to resolve the information in the memory space to the linkage section of the COBOL program.

8. The system of Claim 1, wherein the memory space is operable for a message queue.

9. The system of Claim 8, further comprising a coordination module operable to receive a request from the COBOL program to read and write information to the message queue.

10. The system of Claim 9, wherein the coordination module coordinates reading and writing information to the message queue in a last-in-first-out order.

11. The system of Claim 9, wherein the coordination module coordinates reading and writing information to the message queue in a first-in-first-out order.

12. The system of Claim 1, wherein the memory space is operable for a memory queue.

13. The system of Claim 12, further comprising a coordination module is operable to prevent a conflict.

14. The system of Claim 13, wherein the coordination module is operable from a call to an operating system.

15. The system of Claim 13, wherein the coordination module is operable to prevent writing when the memory space is full and further operable to prevent reading when the memory space is empty.

16. A method of enabling queues for COBOL programs, comprising:

creating a queue using a memory space;

providing an operating system having a key related to an address of the memory space;

maintaining the key in an index;

communicating with the operating system to receive the address of the memory space based on the key;

resolving the memory space to an operable portion of the COBOL program based on the key from the index.

17. The method of Claim 16, wherein a COBOL routine maintains the index and receives the address of the memory space from the operating system.

18. The method of Claim 17, wherein the COBOL routine is further defined as a COBOL technical layer having a plurality of routines, the method further comprising:

attaching to an existing queue;

querying the queue to determine whether the queue exists and to determine the size of the queue;

adding, by a push module of the COBOL technical layer, at least one row to the queue;

blocking when the queue is full;

removing, by a pop module of the COBOL technical layer, a top row from the queue;

detaching from a queue; and

removing a queue from a system.

19. The method of Claim 18, wherein the COBOL technical layer is further defined as a COBOL library wherein the routines are callable from the COBOL program.

20. The method of Claim 18, wherein the COBOL technical layer is integral to the COBOL program.

21. The method of Claim 18, wherein the COBOL technical layer is further defined as enabled by a COBOL compiler.

22. The method of Claim 21, wherein the compiler enabled functionality is further defined as pre-compiler enabled functionality.

23. A method of sharing memory between COBOL programs, comprising:

requesting, by a first and second COBOL programs, an address of a block of memory;

returning the address to the first and second COBOL programs; and

sharing, by the first and second COBOL programs, the block of memory.

24. The method of Claim 1, further comprising allocating the block of shared memory.

25. The method of Claim 24, wherein an operating system allocates the block of shared memory based on parameters obtained from the first and second COBOL programs at compilation.

26. The method of Claim 24, wherein the operating system maintains the address of the block of shared memory and provides the address in response to the first and second COBOL programs requesting the address.

27. The method of Claim 23, wherein the first and second COBOL programs request the address of the block of memory using an identifier.

28. The method of Claim 27, further comprising an index maintaining the identifier associated with a key, the key used to obtain the address of the block of memory from an operating system.

29. The method of Claim 23, wherein the first and second COBOL programs use the address to map a linkage section of each of the first and second COBOL programs to the block of memory to enable the first and second COBOL programs to share the block of memory.

30. A method of sharing memory between COBOL programs, the method comprising:

maintaining, by a COBOL routine, an index of shared memory addresses;

requesting, by a COBOL program, a shared memory block; and

receiving to a linkage section of the COBOL program an address of the shared memory block from the COBOL routine.

31. The method of Claim 30, wherein the COBOL routine is a part of at least the COBOL program.

32. The method of Claim 30, wherein the COBOL routine further comprises a plurality of COBOL subroutines.

33. The method of Claim 30, wherein the COBOL routine is a library routine callable from the COBOL program.

34. The method of Claim 30, wherein the COBOL routine is a COBOL function enabled by a COBOL compiler.

35. The method of Claim 30, further comprising creating, by the COBOL routine, a shared memory block.

36. The method of Claim 35, wherein creating the shared memory block further comprises calling the operating system to from the COBOL routine to request a block of memory.

37. The method of Claim 36, further comprising attaching the COBOL program to the shared memory block.

38. The method of Claim 30, wherein the method further comprises maintaining, by the COBOL routine, an index having an identifier associated with the address of the shared memory block.

39. The method of Claim 38, wherein the method further comprises searching the index based on the identifier to locate the address of the shared memory block associated with the identifier.

40. The method of Claim 39, wherein the searching is accomplished by the COBOL routine in response to receiving a request from the COBOL program, the request including the identifier.

41. The method of Claim 40, wherein the searching is accomplished by the COBOL program.

42. The method of Claim 41, wherein the shared memory is defined as protected, and the method further comprises calling a semaphore routine to manage modifications to the shared memory.

43. The method of Claim 30, further comprising:

creating the shared memory;

attaching to the shared memory;

detaching from the shared memory;

removing the shared memory; and

querying the shared memory to determine whether the shared memory exists and a size of the shared memory.